In [1]:

```
#ANAMAY KRISHNA TIWARI
#20BEE0273
```

5.Handling Missing Values

In [3]:

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

In [4]:

```
data = pd.read_csv('titanic.csv')
```

In [6]:

```
data.head()
```

Out[6]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	d
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False	
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	

In [7]:

```
data.tail()
```

Out[7]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male
886	0	2	male	27.0	0	0	13.00	S	Second	man	True
887	1	1	female	19.0	0	0	30.00	S	First	woman	False
888	0	3	female	NaN	1	2	23.45	S	Third	woman	False
889	1	1	male	26.0	0	0	30.00	C	First	man	True
890	0	3	male	32.0	0	0	7.75	Q	Third	man	True

In [11]:

```
data.sample(5)
```

Out[11]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_ma
304	0	3	male	NaN	0	0	8.0500	S	Third	man	Tru
586	0	2	male	47.0	0	0	15.0000	S	Second	man	Tru
606	0	3	male	30.0	0	0	7.8958	S	Third	man	Tru
362	0	3	female	45.0	0	1	14.4542	C	Third	woman	Fals
193	1	2	male	3.0	1	1	26.0000	S	Second	child	Fals

In [13]:

```
data.shape
```

Out[13]:

```
(891, 15)
```

In [14]:

```
print('Number of columns',data.shape[0])
print('Number of rows',data.shape[1])
```

```
Number of columns 891
Number of rows 15
```

In [16]:

```
data.isnull().sum()
```

Out[16]:

```
survived         0
pclass           0
sex              0
age            177
sibsp            0
parch            0
fare             0
embarked         2
class            0
who              0
adult_male       0
deck           688
embark_town      2
alive            0
alone            0
dtype: int64
```

In [29]:

```
data.isnull().sum().sum()
```

Out[29]:

```
869
```

In [48]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 15 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   survived     891 non-null    int64
 1   pclass       891 non-null    int64
 2   sex          891 non-null    object
 3   age          714 non-null    float64
 4   sibsp        891 non-null    int64
 5   parch        891 non-null    int64
 6   fare         891 non-null    float64
 7   embarked     889 non-null    object
 8   class        891 non-null    object
 9   who          891 non-null    object
 10  adult_male   891 non-null    bool
 11  deck         203 non-null    object
 12  embark_town  889 non-null    object
 13  alive        891 non-null    object
 14  alone        891 non-null    bool
dtypes: bool(2), float64(2), int64(4), object(7)
memory usage: 92.4+ KB
```

In [33]:

```
df = data.fillna(value=0)
df
```

Out[33]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_ma
0	0	3	male	22.0	1	0	7.2500	S	Third	man	Tru
1	1	1	female	38.0	1	0	71.2833	C	First	woman	Fals
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	Fals
3	1	1	female	35.0	1	0	53.1000	S	First	woman	Fals
4	0	3	male	35.0	0	0	8.0500	S	Third	man	Tru
...	...	...	...	...	...	...	...	...	...	...	
886	0	2	male	27.0	0	0	13.0000	S	Second	man	Tru
887	1	1	female	19.0	0	0	30.0000	S	First	woman	Fals
888	0	3	female	0.0	1	2	23.4500	S	Third	woman	Fals
889	1	1	male	26.0	0	0	30.0000	C	First	man	Tru
890	0	3	male	32.0	0	0	7.7500	Q	Third	man	Tru

891 rows × 15 columns

In [46]:

```
df.isnull().sum()
```

Out[46]:

```
survived       0
pclass         0
sex            0
age            0
sibsp          0
parch          0
fare           0
embarked       0
class          0
who            0
adult_male     0
deck           0
embark_town    0
alive          0
alone          0
dtype: int64
```

In [47]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 15 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   survived     891 non-null    int64
 1   pclass       891 non-null    int64
 2   sex          891 non-null    object
 3   age          891 non-null    float64
 4   sibsp        891 non-null    int64
 5   parch        891 non-null    int64
 6   fare         891 non-null    float64
 7   embarked     891 non-null    object
 8   class        891 non-null    object
 9   who          891 non-null    object
 10  adult_male   891 non-null    bool
 11  deck         891 non-null    object
 12  embark_town  891 non-null    object
 13  alive        891 non-null    object
 14  alone        891 non-null    bool
dtypes: bool(2), float64(2), int64(4), object(7)
memory usage: 92.4+ KB
```

In [50]:

```
df.describe()
```

Out[50]:

	survived	pclass	age	sibsp	parch	fare
count	891.000000	891.000000	891.000000	891.000000	891.000000	891.000000
mean	0.383838	2.308642	23.799293	0.523008	0.381594	32.204208
std	0.486592	0.836071	17.596074	1.102743	0.806057	49.693429
min	0.000000	1.000000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	2.000000	6.000000	0.000000	0.000000	7.910400
50%	0.000000	3.000000	24.000000	0.000000	0.000000	14.454200
75%	1.000000	3.000000	35.000000	1.000000	0.000000	31.000000
max	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

In [200]:

```
df1 = data.fillna(method='pad')
df1
```

Out[200]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_ma
0	0	3	male	22.0	1	0	7.2500	S	Third	man	Tru
1	1	1	female	38.0	1	0	71.2833	C	First	woman	Fals
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	Fals
3	1	1	female	35.0	1	0	53.1000	S	First	woman	Fals
4	0	3	male	35.0	0	0	8.0500	S	Third	man	Tru
...	...	...	...	...	...	...	...	...	...	...	
886	0	2	male	27.0	0	0	13.0000	S	Second	man	Tru
887	1	1	female	19.0	0	0	30.0000	S	First	woman	Fals
888	0	3	female	19.0	1	2	23.4500	S	Third	woman	Fals
889	1	1	male	26.0	0	0	30.0000	C	First	man	Tru
890	0	3	male	32.0	0	0	7.7500	Q	Third	man	Tru

891 rows × 15 columns

UNIVARIATE ANALYSIS

In [54]:

```
cat = []
num = []
for column in data.columns:
    if df[column].nunique() > 10:
        num.append(column)
    else:
        cat.append(column)
```

In [55]:

```
cat
```

Out[55]:

```
['survived',
 'pclass',
 'sex',
 'sibsp',
 'parch',
 'embarked',
 'class',
 'who',
 'adult_male',
 'deck',
 'embark_town',
 'alive',
 'alone']
```

In [56]:

```
num
```

Out[56]:

```
['age', 'fare']
```

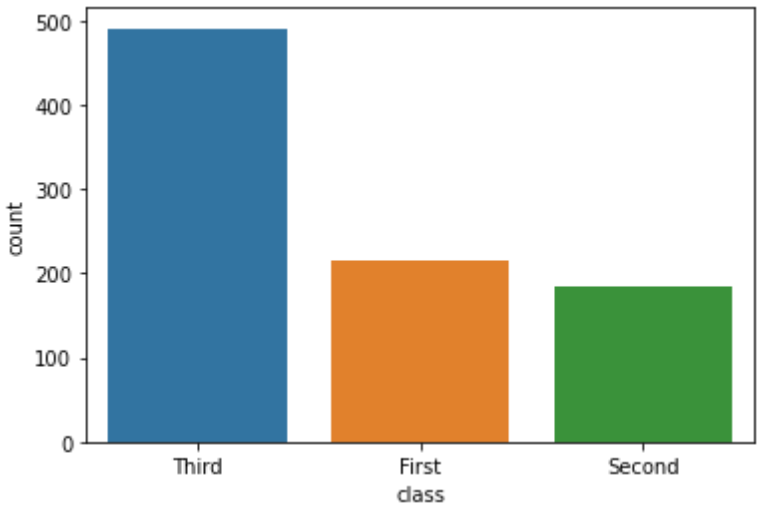
In [58]:

```
sns.countplot(df['class'])
```

```
C:\Users\Anamay Tiwari\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other ar
guments without an explicit keyword will result in an error or misinterpretat
ion.
  warnings.warn(
```

Out[58]:

```
<AxesSubplot:xlabel='class', ylabel='count'>
```



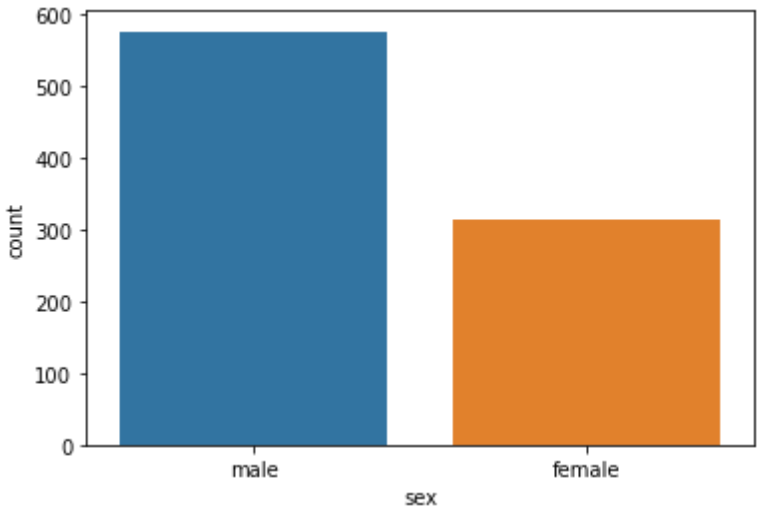
In [59]:

```
sns.countplot(df['sex'])
```

```
C:\Users\Anamay Tiwari\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other ar
guments without an explicit keyword will result in an error or misinterpretat
ion.
  warnings.warn(
```

Out[59]:

```
<AxesSubplot:xlabel='sex', ylabel='count'>
```



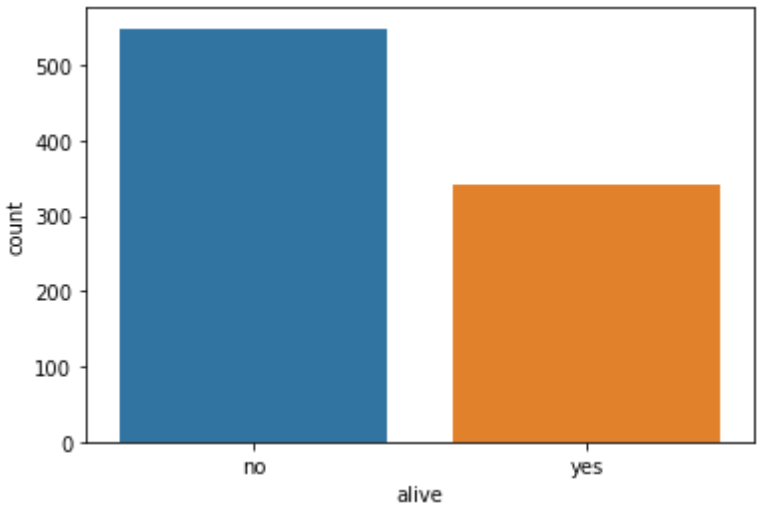
In [61]:

```
sns.countplot(df['alive'])
```

```
C:\Users\Anamay Tiwari\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other ar
guments without an explicit keyword will result in an error or misinterpretat
ion.
  warnings.warn(
```

Out[61]:

```
<AxesSubplot:xlabel='alive', ylabel='count'>
```



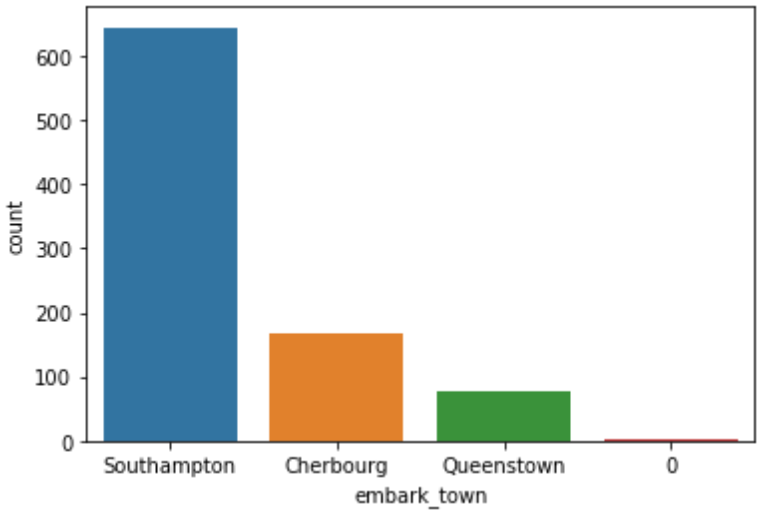
In [63]:

```
sns.countplot(df['embark_town'])
```

```
C:\Users\Anamay Tiwari\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other ar
guments without an explicit keyword will result in an error or misinterpretat
ion.
  warnings.warn(
```

Out[63]:

```
<AxesSubplot:xlabel='embark_town', ylabel='count'>
```

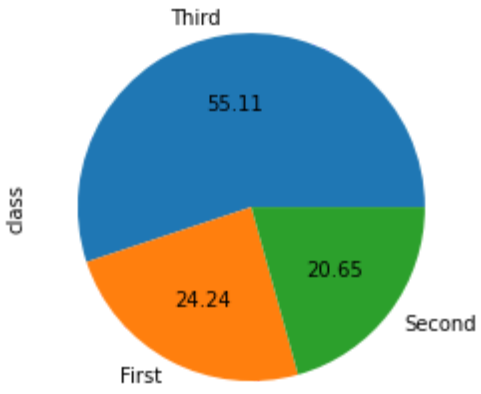


In [67]:

```
df['class'].value_counts().plot(kind='pie',autopct="%1.2f")
```

Out[67]:

```
<AxesSubplot:ylabel='class'>
```

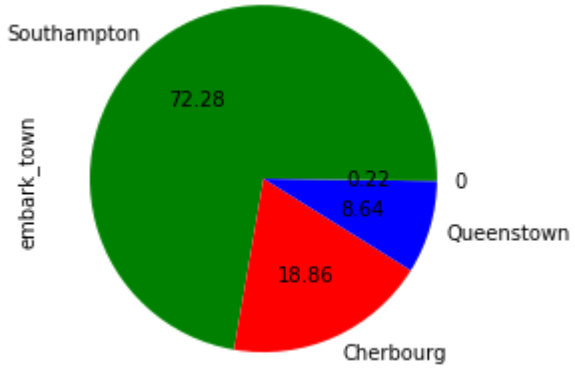


In [70]:

```
df['embark_town'].value_counts().plot(kind='pie',autopct="%1.2f",colors=['green','red','blu
```

Out[70]:

```
<AxesSubplot:ylabel='embark_town'>
```



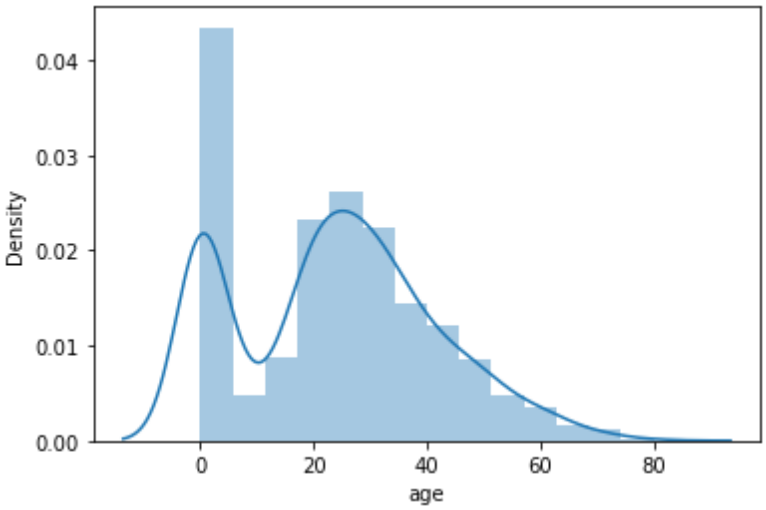
In [73]:

```
sns.distplot(df['age'])
```

```
C:\Users\Anamay Tiwari\anaconda3\lib\site-packages\seaborn\distributions.py:2
557: FutureWarning: `distplot` is a deprecated function and will be removed i
n a future version. Please adapt your code to use either `displot` (a figure-
level function with similar flexibility) or `histplot` (an axes-level functio
n for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[73]:

```
<AxesSubplot:xlabel='age', ylabel='Density'>
```



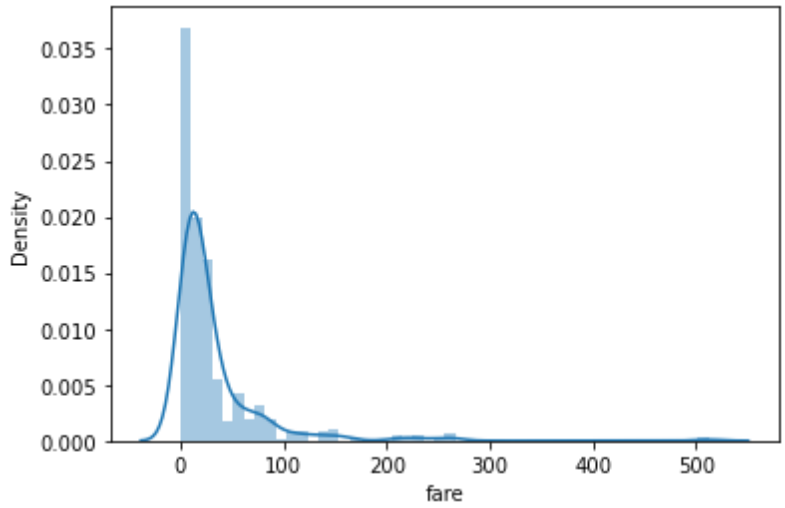
In [75]:

```
sns.distplot(df['fare'])
```

```
C:\Users\Anamay Tiwari\anaconda3\lib\site-packages\seaborn\distributions.py:2
557: FutureWarning: `distplot` is a deprecated function and will be removed i
n a future version. Please adapt your code to use either `displot` (a figure-
level function with similar flexibility) or `histplot` (an axes-level functio
n for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[75]:

```
<AxesSubplot:xlabel='fare', ylabel='Density'>
```



In [74]:

```
df['age'].skew()
```

Out[74]:

```
0.2628619929342128
```

In [76]:

```
df['fare'].skew()
```

Out[76]:

```
4.787316519674893
```

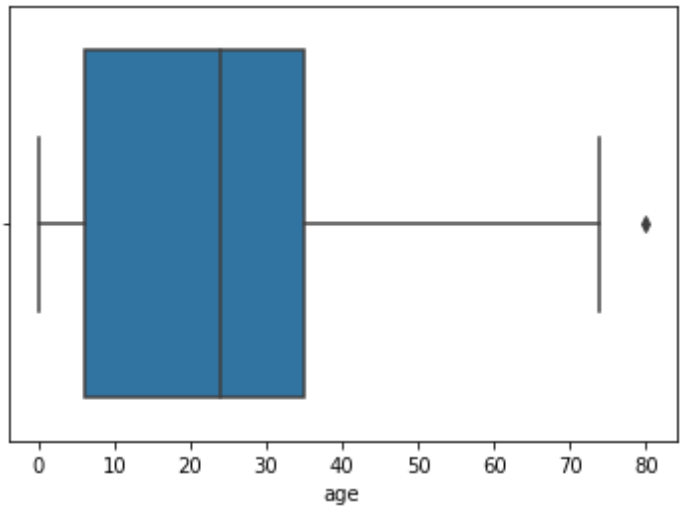
In [80]:

```
sns.boxplot(df['age'])
```

```
C:\Users\Anamay Tiwari\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other ar
guments without an explicit keyword will result in an error or misinterpretat
ion.
  warnings.warn(
```

Out[80]:

```
<AxesSubplot:xlabel='age'>
```



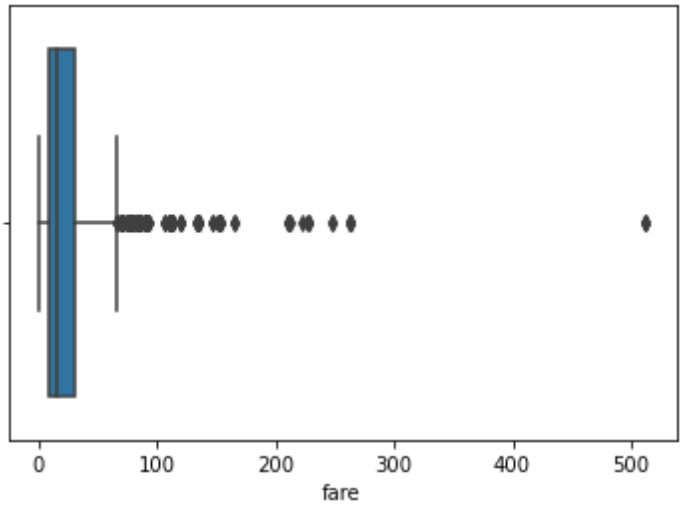
In [42]:

```
y = sns.boxplot(data.fare)
y
```

```
C:\Users\Anamay Tiwari\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other ar
guments without an explicit keyword will result in an error or misinterpretat
ion.
  warnings.warn(
```

Out[42]:

```
<AxesSubplot:xlabel='fare'>
```



Bi-Variate Analysis

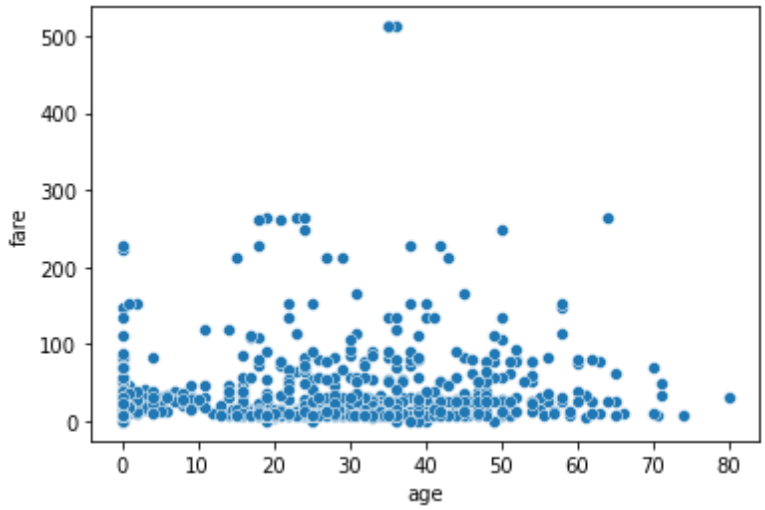
In [90]:

```
sns.scatterplot(df['age'],df['fare'])
```

```
C:\Users\Anamay Tiwari\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variables as keyword args: x, y. From versi
on 0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or misinterpret
ation.
  warnings.warn(
```

Out[90]:

```
<AxesSubplot:xlabel='age', ylabel='fare'>
```



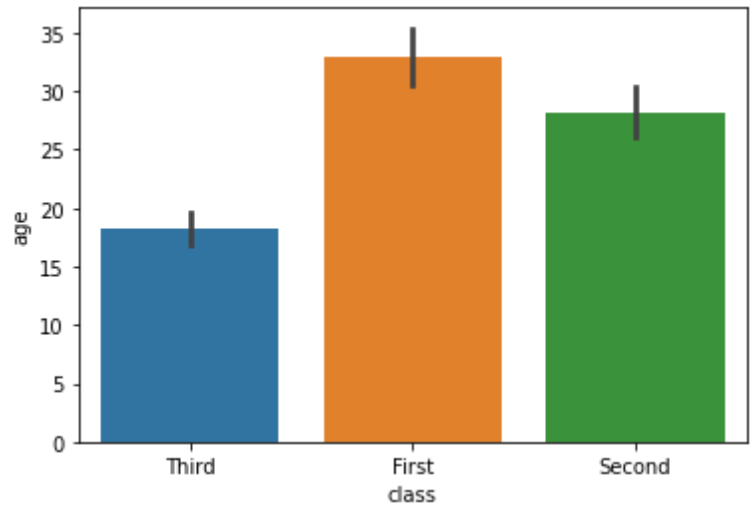
In [91]:

```
sns.barplot(df['class'],df['age'])
```

```
C:\Users\Anamay Tiwari\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variables as keyword args: x, y. From versi
on 0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or misinterpret
ation.
  warnings.warn(
```

Out[91]:

```
<AxesSubplot:xlabel='class', ylabel='age'>
```



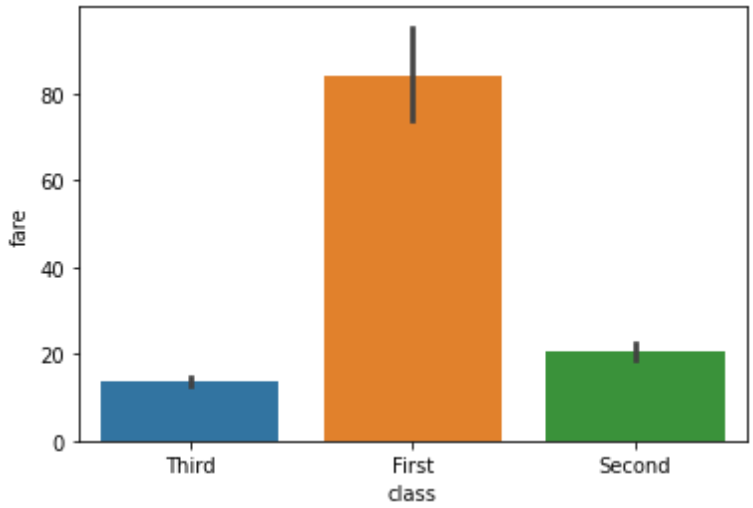
In [92]:

```
sns.barplot(df['class'],df['fare'])
```

```
C:\Users\Anamay Tiwari\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variables as keyword args: x, y. From versi
on 0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or misinterpret
ation.
  warnings.warn(
```

Out[92]:

```
<AxesSubplot:xlabel='class', ylabel='fare'>
```

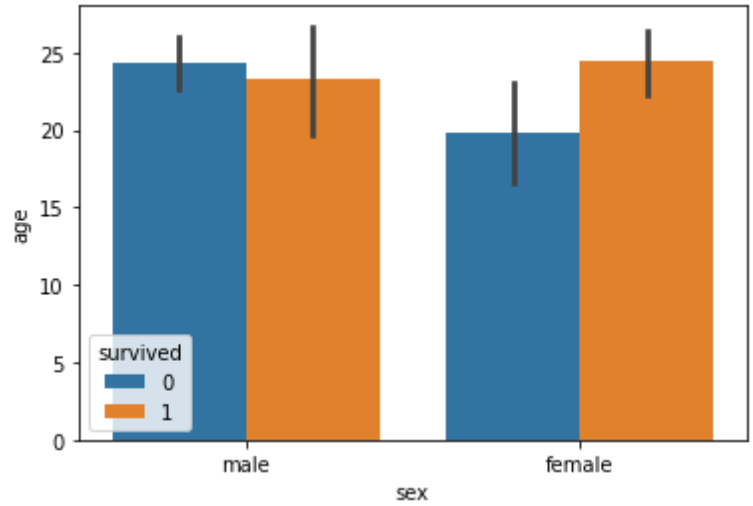


In [128]:

```
sns.barplot(df['sex'],df['age'],hue=df['survived'])
```

Out[128]:

```
<AxesSubplot:xlabel='sex', ylabel='age'>
```



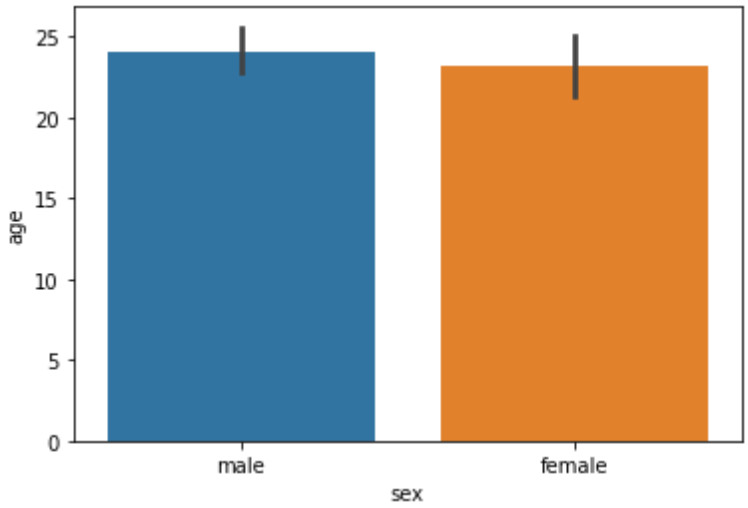
In [129]:

```
sns.barplot(df['sex'],df['age'])
```

```
C:\Users\Anamay Tiwari\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variables as keyword args: x, y. From versi
on 0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or misinterpret
ation.
  warnings.warn(
```

Out[129]:

```
<AxesSubplot:xlabel='sex', ylabel='age'>
```



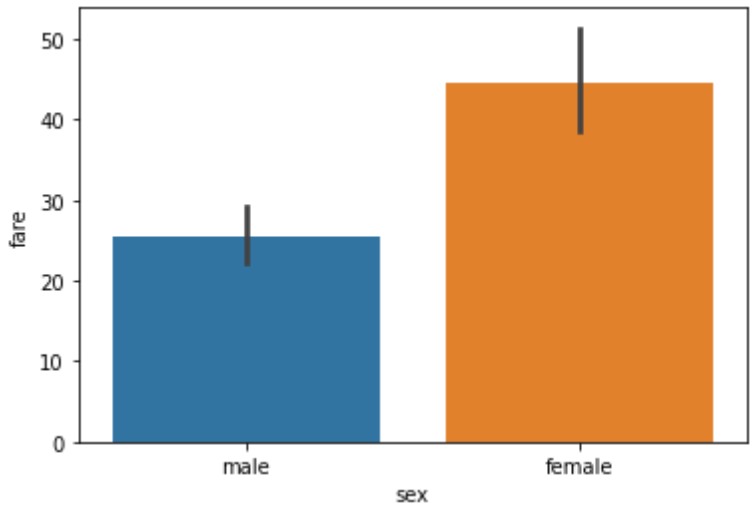
In [99]:

```
sns.barplot(df['sex'],df['fare'])
```

```
C:\Users\Anamay Tiwari\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variables as keyword args: x, y. From versi
on 0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or misinterpret
ation.
  warnings.warn(
```

Out[99]:

```
<AxesSubplot:xlabel='sex', ylabel='fare'>
```



In [102]:

```
cat
```

Out[102]:

```
['survived',
 'pclass',
 'sex',
 'sibsp',
 'parch',
 'embarked',
 'class',
 'who',
 'adult_male',
 'deck',
 'embark_town',
 'alive',
 'alone']
```

In [103]:

```
num
```

Out[103]:

```
['age', 'fare']
```

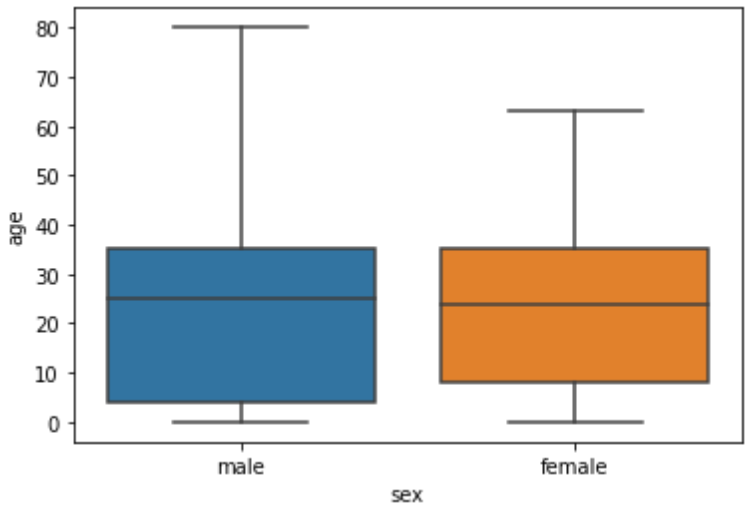
In [105]:

```
sns.boxplot(df['sex'],df['age'])
```

```
C:\Users\Anamay Tiwari\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variables as keyword args: x, y. From versi
on 0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or misinterpret
ation.
  warnings.warn(
```

Out[105]:

```
<AxesSubplot:xlabel='sex', ylabel='age'>
```



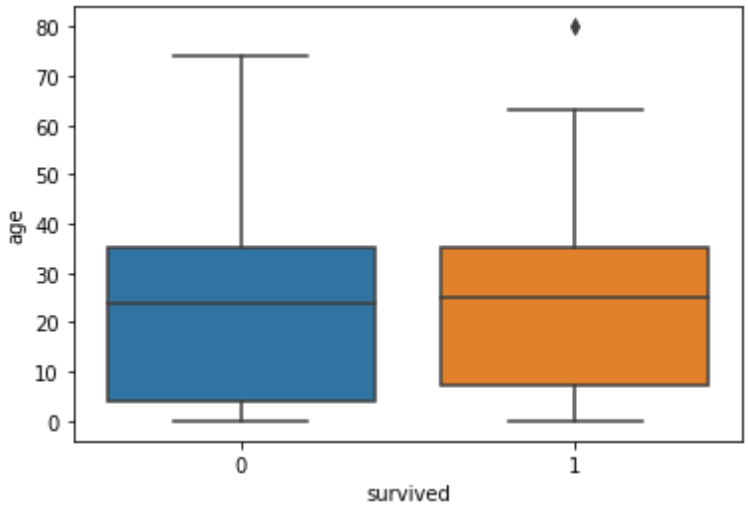
In [108]:

```
sns.boxplot(df['survived'],df['age'])
```

```
C:\Users\Anamay Tiwari\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variables as keyword args: x, y. From versi
on 0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or misinterpret
ation.
  warnings.warn(
```

Out[108]:

```
<AxesSubplot:xlabel='survived', ylabel='age'>
```



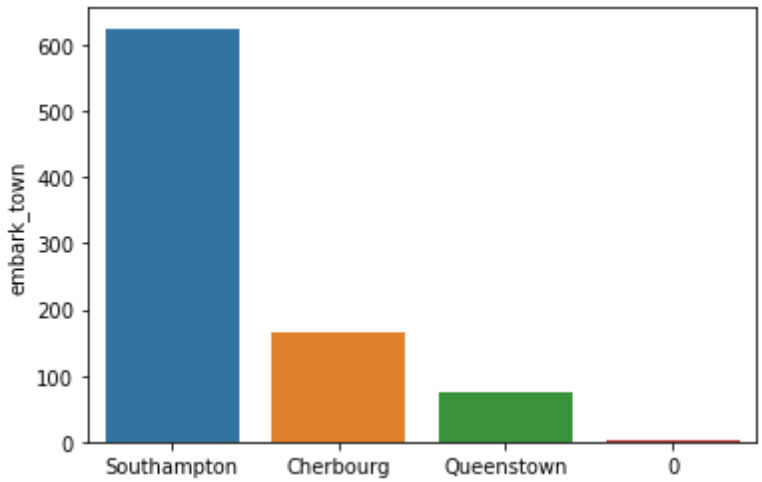
In [255]:

```
sns.barplot((df.embark_town.value_counts()).index,df.embark_town.value_counts())
```

```
C:\Users\Anamay Tiwari\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variables as keyword args: x, y. From versi
on 0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or misinterpret
ation.
  warnings.warn(
```

Out[255]:

```
<AxesSubplot:ylabel='embark_town'>
```



In [257]:

```
sns.barplot((df.class.value_counts()).index,df.class.value_counts())
```

```
  File "<ipython-input-257-b7847e5d23f7>", line 1
    sns.barplot((df.class.value_counts()).index,df.class.value_counts())
                    ^
SyntaxError: invalid syntax
```

Multi-Variate Analysis

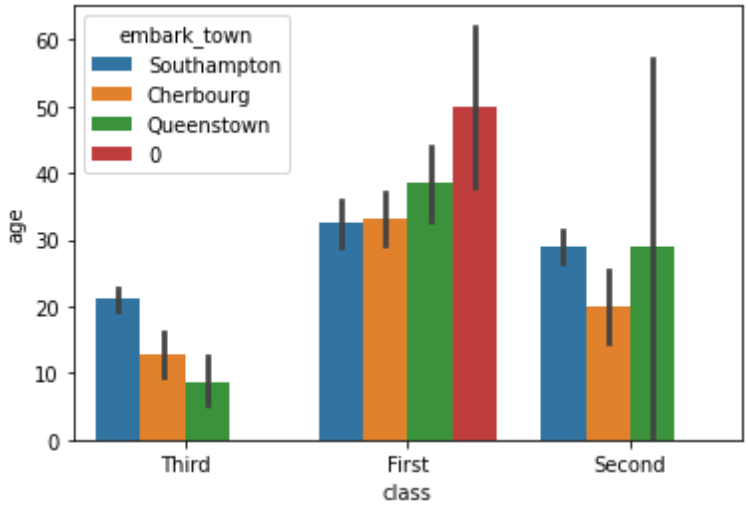
In [100]:

```
sns.barplot(df['class'],df['age'],hue=df['embark_town'])
```

```
C:\Users\Anamay Tiwari\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variables as keyword args: x, y. From versi
on 0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or misinterpret
ation.
  warnings.warn(
```

Out[100]:

```
<AxesSubplot:xlabel='class', ylabel='age'>
```



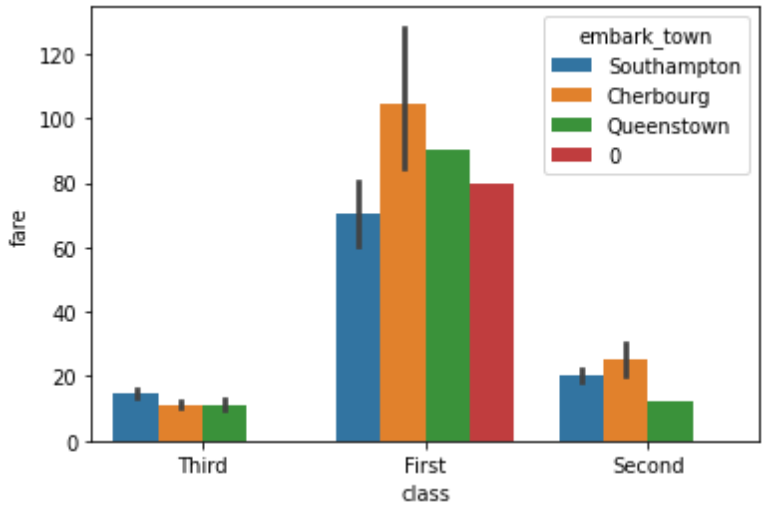
In [101]:

```
sns.barplot(df['class'],df['fare'],hue=df['embark_town'])
```

```
C:\Users\Anamay Tiwari\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variables as keyword args: x, y. From versi
on 0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or misinterpret
ation.
  warnings.warn(
```

Out[101]:

```
<AxesSubplot:xlabel='class', ylabel='fare'>
```



In [110]:

```
df.columns
```

Out[110]:

```
Index(['survived', 'pclass', 'sex', 'age', 'sibsp', 'parch', 'fare',
       'embarked', 'class', 'who', 'adult_male', 'deck', 'embark_town',
       'alive', 'alone'],
      dtype='object')
```

In [117]:

```
cat
```

Out[117]:

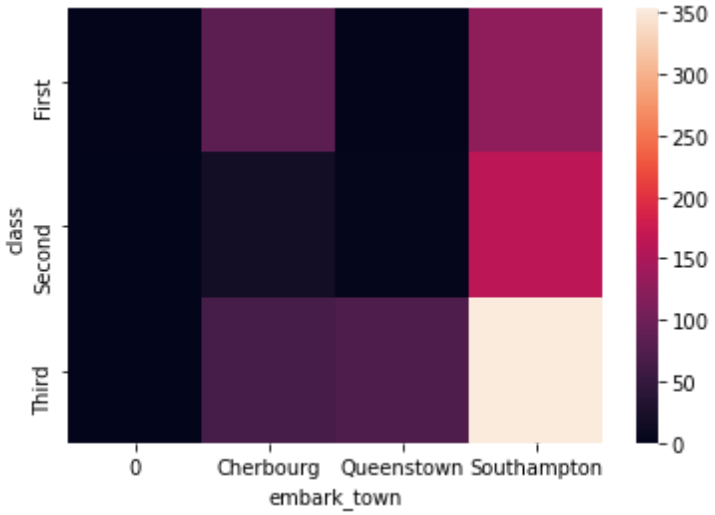
```
['survived',
 'pclass',
 'sex',
 'sibsp',
 'parch',
 'embarked',
 'class',
 'who',
 'adult_male',
 'deck',
 'embark_town',
 'alive',
 'alone']
```

In [122]:

```
sns.heatmap(pd.crosstab(df['class'],df['embark_town']))
```

Out[122]:

<AxesSubplot:xlabel='embark_town', ylabel='class'>

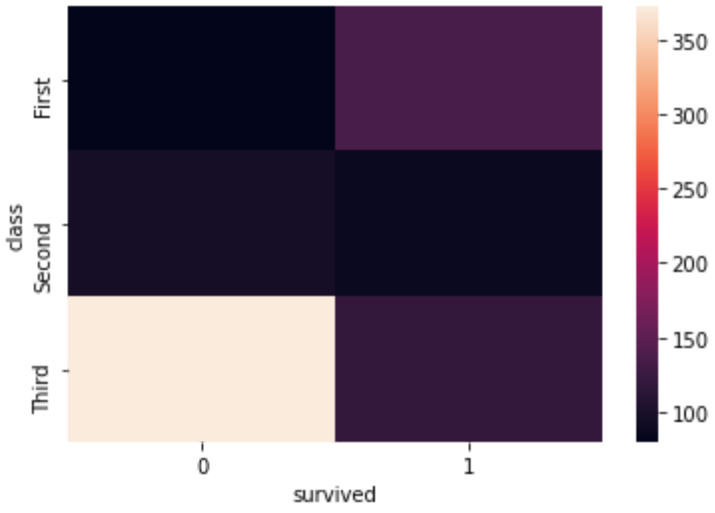


In [124]:

```
sns.heatmap(pd.crosstab(df['class'],df['survived']))
```

Out[124]:

<AxesSubplot:xlabel='survived', ylabel='class'>



4.DESCRIPTIVE STATISTICS

In [147]:

```
df.mean()
```

Out[147]:

```
survived       0.383838
pclass         2.308642
age           23.799293
sibsp          0.523008
parch          0.381594
fare          32.204208
adult_male     0.602694
alone          0.602694
dtype: float64
```

In [151]:

```
df.median()
```

Out[151]:

```
survived       0.0000
pclass         3.0000
age           24.0000
sibsp          0.0000
parch          0.0000
fare          14.4542
adult_male     1.0000
alone          1.0000
dtype: float64
```

In [150]:

```
df.mode()
```

Out[150]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	er
0	0	3	male	0.0	0	0	8.05	S	Third	man	True	0	S

In [177]:

```
df.skew()
```

Out[177]:

```
survived      0.478523
pclass       -0.630548
age           0.262862
sibsp         3.695352
parch         2.749117
fare          4.787317
adult_male   -0.420431
alone        -0.420431
dtype: float64
```

In [178]:

```
#kurtosis
df.kurt()
```

Out[178]:

```
survived      -1.775005
pclass        -1.280015
age           -0.537533
sibsp         17.880420
parch          9.778125
fare          33.398141
adult_male    -1.827345
alone         -1.827345
dtype: float64
```

In [179]:

```
#range
df.max()
```

Out[179]:

```
survived            1
pclass              3
sex              male
age              80.0
sibsp               8
parch               6
fare          512.3292
class            Third
who              woman
adult_male        True
alive              yes
alone             True
dtype: object
```

In [180]:

```
df.min()
```

Out[180]:

```
survived          0
pclass            1
sex          female
age             0.0
sibsp             0
parch             0
fare            0.0
class         First
who           child
adult_male    False
alive            no
alone         False
dtype: object
```

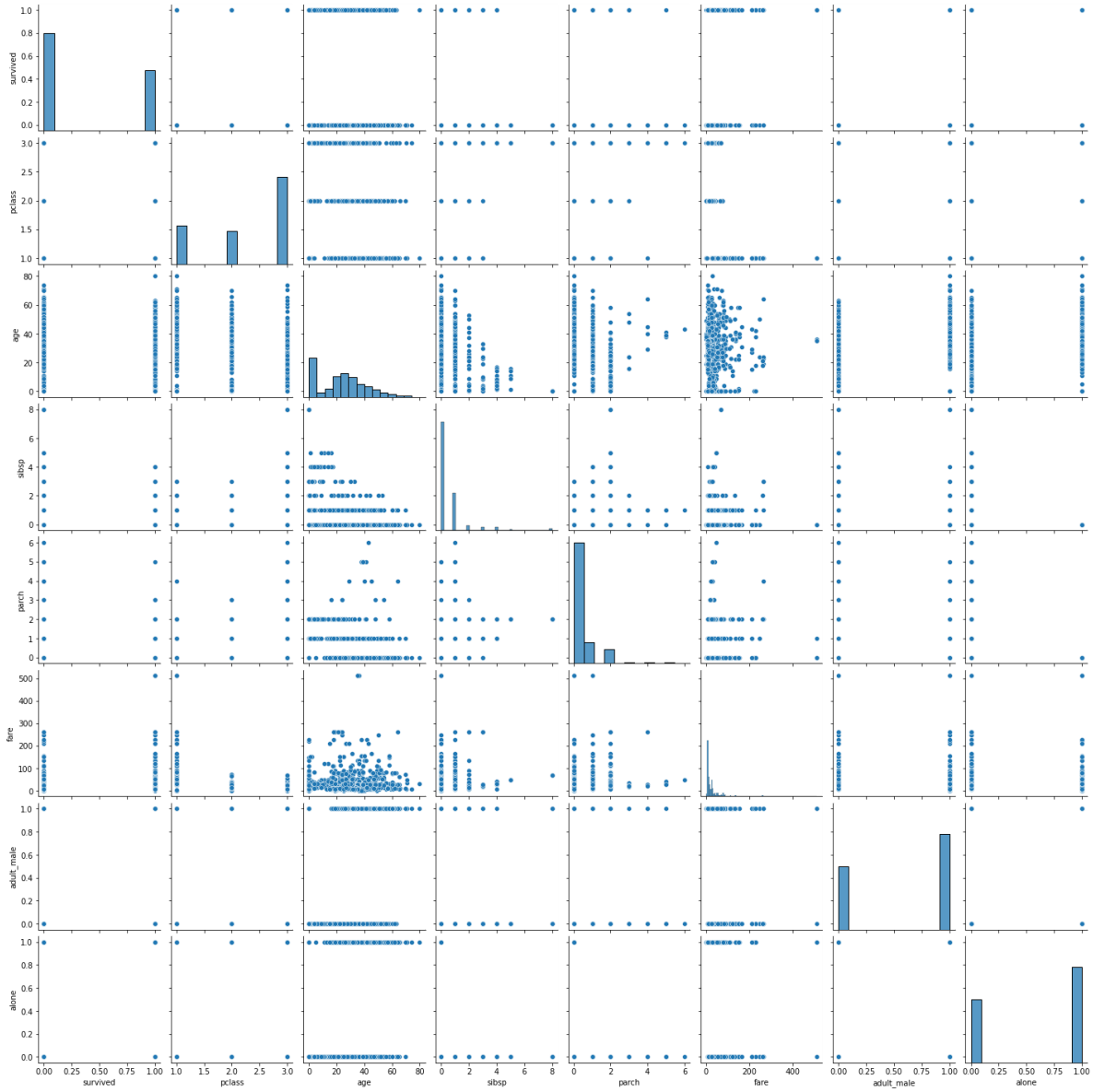
In [186]:

```
sns.pairplot(df)
```

```
<__array_function__ internals>:5: RuntimeWarning: Converting input from bool
to <class 'numpy.uint8'> for compatibility.
<__array_function__ internals>:5: RuntimeWarning: Converting input from bool
to <class 'numpy.uint8'> for compatibility.
<__array_function__ internals>:5: RuntimeWarning: Converting input from bool
to <class 'numpy.uint8'> for compatibility.
<__array_function__ internals>:5: RuntimeWarning: Converting input from bool
to <class 'numpy.uint8'> for compatibility.
```

Out[186]:

```
<seaborn.axisgrid.PairGrid at 0x1d0e4a9d3d0>
```



In [170]:

```
num
```

Out[170]:

```
['age', 'fare']
```

In [193]:

```
df.var()
```

Out[193]:

```
survived         0.236772
pclass           0.699015
age            309.621823
sibsp            1.216043
parch            0.649728
fare          2469.436846
adult_male       0.239723
alone            0.239723
dtype: float64
```

In [194]:

```
df.std()
```

Out[194]:

```
survived       0.486592
pclass         0.836071
age           17.596074
sibsp          1.102743
parch          0.806057
fare          49.693429
adult_male     0.489615
alone          0.489615
dtype: float64
```

In [195]:

```
df['sex'].value_counts()
```

Out[195]:

```
male      577
female    314
Name: sex, dtype: int64
```

In [196]:

```
data['embark_town'].value_counts()
```

Out[196]:

```
Southampton    644
Cherbourg      168
Queenstown      77
Name: embark_town, dtype: int64
```

In [197]:

```
data['alive'].value_counts()
```

Out[197]:

```
no     549
yes    342
Name: alive, dtype: int64
```

In [198]:

```
data['class'].value_counts()
```

Out[198]:

```
Third     491
First     216
Second    184
Name: class, dtype: int64
```

In [202]:

```
df.isnull().any()
```

Out[202]:

```
survived       False
pclass         False
sex            False
age            False
sibsp          False
parch          False
fare           False
embarked       False
class          False
who            False
adult_male     False
deck           False
embark_town    False
alive          False
alone          False
dtype: bool
```

8.Dependent Variable

In [205]:

```
x = df.iloc[:,0:1]
x
```

Out[205]:

	survived
0	0
1	1
2	1
3	1
4	0
...	...
886	0
887	1
888	0
889	1
890	0

891 rows × 1 columns

8.Independent Variable

In [206]:

```
y = df.iloc[:,1:]
y
```

Out[206]:

	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck
0	3	male	22.0	1	0	7.2500	S	Third	man	True	0
1	1	female	38.0	1	0	71.2833	C	First	woman	False	C
2	3	female	26.0	0	0	7.9250	S	Third	woman	False	0
3	1	female	35.0	1	0	53.1000	S	First	woman	False	C
4	3	male	35.0	0	0	8.0500	S	Third	man	True	0
...	...	...	...	...	...	...	...	...	...	...	...
886	2	male	27.0	0	0	13.0000	S	Second	man	True	0
887	1	female	19.0	0	0	30.0000	S	First	woman	False	B
888	3	female	0.0	1	2	23.4500	S	Third	woman	False	0
889	1	male	26.0	0	0	30.0000	C	First	man	True	C
890	3	male	32.0	0	0	7.7500	Q	Third	man	True	0

891 rows × 14 columns

6.Outlier Detection

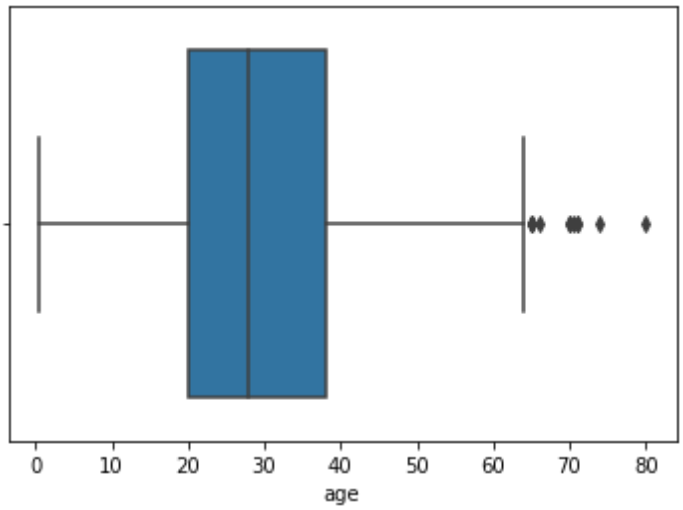
In [232]:

```
sns.boxplot(data.age)
```

```
C:\Users\Anamay Tiwari\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other ar
guments without an explicit keyword will result in an error or misinterpretat
ion.
  warnings.warn(
```

Out[232]:

```
<AxesSubplot:xlabel='age'>
```



In [234]:

```
prec99 = data.age.quantile(0.99)
prec99
```

Out[234]:

```
65.87
```

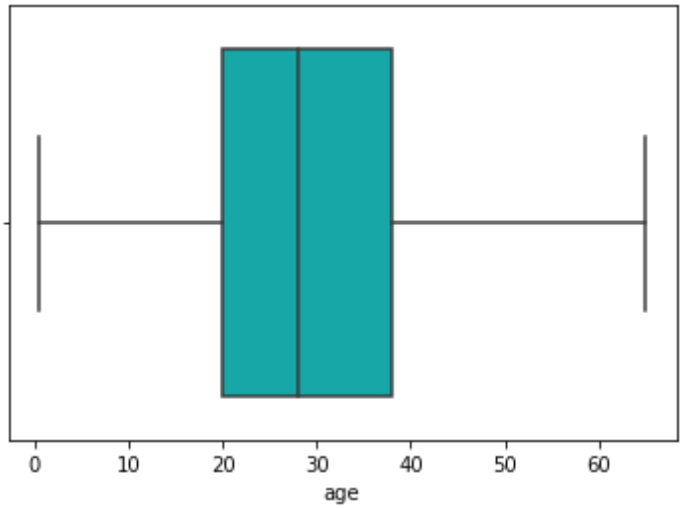
In [254]:

```
data = data[data.age<=prec99]
sns.boxplot(data.age,color='c')
```

```
C:\Users\Anamay Tiwari\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other ar
guments without an explicit keyword will result in an error or misinterpretat
ion.
  warnings.warn(
```

Out[254]:

```
<AxesSubplot:xlabel='age'>
```



In [258]:

```
df.head()
```

Out[258]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False
27	0	1	male	19.0	3	2	263.0000	S	First	man	True
28	1	3	female	0.0	0	0	7.8792	Q	Third	woman	False
29	0	3	male	0.0	0	0	7.8958	S	Third	man	True

7.Encoding

In [269]:

```
#Label Encoding
from sklearn.preprocessing import LabelEncoder
```

In [270]:

```
le = LabelEncoder()
```

In [274]:

```
df.who = le.fit_transform(df.who) #Label Encoded Who column
```

In [275]:

```
df.head()
```

Out[275]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	decl
0	0	3	29	1	1	0	7.2500	S	Third	1	True	(
1	1	1	52	2	1	0	71.2833	C	First	2	False	C
27	0	1	25	1	3	2	263.0000	S	First	1	True	C
28	1	3	0	2	0	0	7.8792	Q	Third	2	False	(
29	0	3	0	1	0	0	7.8958	S	Third	1	True	(

In [279]:

```
df.adult_male = le.fit_transform(df.adult_male)
```

In [284]:

```
df.alive = le.fit_transform(df.alive)
```

In [286]:

```
df.alone = le.fit_transform(df.alone)
```

In [310]:

```
df.deck =  le.fit_transform(df.deck)
```

In [321]:

```
df.embarked = le.fit_transform(df.alone)
```

In [322]:

```
df.head()
```

Out[322]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	decl
0	0	3	29	1	1	0	7.2500	0	Third	1	1	(
1	1	1	52	2	1	0	71.2833	0	First	2	2	(
27	0	1	25	1	3	2	263.0000	0	First	1	1	(
28	1	3	0	2	0	0	7.8792	1	Third	2	2	
29	0	3	0	1	0	0	7.8958	1	Third	1	1	

In [336]:

```
#One Hot encoding
df_main = pd.get_dummies(df,columns=['embark_town'])
```

In [337]:

```
df_main.head()
```

Out[337]:

	fare	embarked	class	who	adult_male	deck	alive	alone	embark_town_0	embark_town_Che
	7.2500	0	Third	1	1	0	0	0	0	
	71.2833	0	First	2	2	0	1	0	0	
	263.0000	0	First	1	1	0	0	0	0	
	7.8792	1	Third	2	2	1	1	1	0	
	7.8958	1	Third	1	1	1	0	1	0	

In [338]:

```
df_main.corr()
```

Out[338]:

	survived	pclass	sex	age	sibsp	parch	
survived	1.000000	-0.340729	0.014421	0.327997	-0.027969	0.084976	0.26
pclass	-0.340729	1.000000	-0.356028	-0.194439	0.077995	0.002829	-0.5
sex	0.014421	-0.356028	1.000000	0.185727	-0.170721	-0.043989	0.13
age	0.327997	-0.194439	0.185727	1.000000	-0.123402	-0.062820	0.14
sibsp	-0.027969	0.077995	-0.170721	-0.123402	1.000000	0.424508	0.16
parch	0.084976	0.002829	-0.043989	-0.062820	0.424508	1.000000	0.22
fare	0.263684	-0.551198	0.138338	0.147645	0.160904	0.225947	1.00
embarked	-0.210584	0.145067	0.012029	0.001132	-0.580721	-0.589799	-0.27
who	0.327997	-0.194439	0.185727	1.000000	-0.123402	-0.062820	0.14
adult_male	0.327997	-0.194439	0.185727	1.000000	-0.123402	-0.062820	0.14
deck	-0.210584	0.145067	0.012029	0.001132	-0.580721	-0.589799	-0.27
alive	1.000000	-0.340729	0.014421	0.327997	-0.027969	0.084976	0.26
alone	-0.210584	0.145067	0.012029	0.001132	-0.580721	-0.589799	-0.27
embark_town_0	0.061468	-0.074912	0.070857	0.064241	-0.022681	-0.023036	0.04
embark_town_Cherbourg	0.171535	-0.249716	0.008076	0.031899	-0.057990	-0.002755	0.27
embark_town_Queenstown	0.007547	0.219664	-0.242625	0.112695	-0.035424	-0.082919	-0.11
embark_town_Southampton	-0.161607	0.089828	0.136678	-0.105131	0.075343	0.056604	-0.17

5.Splitting data into Dependent and Independent Variables

In [425]:

```
 # X & Y Split
#Y is dependent variable
y = df_main['survived']
y.head()
```

Out[425]:

```
0       7.2500
1      71.2833
27    263.0000
28      7.8792
29      7.8958
Name: fare, dtype: float64
```

In [463]:

```
X = df_main.drop(columns=['survived','class'],axis=1)
```

In [464]:

```
X.head()
```

Out[464]:

	pclass	sex	age	sibsp	parch	fare	embarked	who	adult_male	deck	alive	alone	e
0	3	29	1	1	0	7.2500	0	1	1	0	0	0	
1	1	52	2	1	0	71.2833	0	2	2	0	1	0	
27	1	25	1	3	2	263.0000	0	1	1	0	0	0	
28	3	0	2	0	0	7.8792	1	2	2	1	1	1	
29	3	0	1	0	0	7.8958	1	1	1	1	0	1	

In [465]:

```
name = X.columns
```

In [466]:

```
name
```

Out[466]:

```
Index(['pclass', 'sex', 'age', 'sibsp', 'parch', 'fare', 'embarked', 'who',
       'adult_male', 'deck', 'alive', 'alone', 'embark_town_0',
       'embark_town_Cherbourg', 'embark_town_Queenstown',
       'embark_town_Southampton'],
      dtype='object')
```

9.Scaling

In [467]:

```
from sklearn.preprocessing import MinMaxScaler
```

In [468]:

```
scale = MinMaxScaler()
```

In [469]:

```
X_scaled = scale.fit_transform(X)
X_scaled
```

Out[469]:

```
array([[1.        , 0.32954545, 0.5       , ..., 0.        , 0.        ,
        1.        ],
       [0.        , 0.59090909, 1.        , ..., 1.        , 0.        ,
        0.        ],
       [0.        , 0.28409091, 0.5       , ..., 0.        , 0.        ,
        1.        ],
       ...,
       [1.        , 0.        , 1.        , ..., 0.        , 0.        ,
        1.        ],
       [0.        , 0.39772727, 0.5       , ..., 1.        , 0.        ,
        0.        ],
       [1.        , 0.48863636, 0.5       , ..., 0.        , 1.        ,
        0.        ]])
```

In [470]:

```
X = pd.DataFrame(X_scaled,columns=name)
X
```

Out[470]:

	pclass	sex	age	sibsp	parch	fare	embarked	who	adult_male	deck	alive
0	1.0	0.329545	0.5	0.125	0.000000	0.014151	0.0	0.5	0.5	0.0	0.0
1	0.0	0.590909	1.0	0.125	0.000000	0.139136	0.0	1.0	1.0	0.0	1.0
2	0.0	0.284091	0.5	0.375	0.333333	0.513342	0.0	0.5	0.5	0.0	0.0
3	1.0	0.000000	1.0	0.000	0.000000	0.015379	1.0	1.0	1.0	1.0	1.0
4	1.0	0.000000	0.5	0.000	0.000000	0.015412	1.0	0.5	0.5	1.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...
861	0.5	0.409091	0.5	0.000	0.000000	0.025374	1.0	0.5	0.5	1.0	0.0
862	0.0	0.284091	1.0	0.000	0.000000	0.058556	1.0	1.0	1.0	1.0	1.0
863	1.0	0.000000	1.0	0.125	0.333333	0.045771	0.0	1.0	1.0	0.0	0.0
864	0.0	0.397727	0.5	0.000	0.000000	0.058556	1.0	0.5	0.5	1.0	1.0
865	1.0	0.488636	0.5	0.000	0.000000	0.015127	1.0	0.5	0.5	1.0	0.0

866 rows × 16 columns

10.Train Test Split

In [471]:

```
from sklearn.model_selection import train_test_split
```

In [472]:

```
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.2,random_state=0)
```

In [473]:

```
X_train.head()
```

Out[473]:

	pclass	sex	age	sibsp	parch	fare	embarked	who	adult_male	deck	alive
711	1.0	0.727273	1.0	0.125	0.500000	0.067096	0.0	1.0	1.0	0.0	0.0
156	0.5	0.000000	0.5	0.000	0.000000	0.029376	1.0	0.5	0.5	1.0	0.0
356	1.0	0.068182	0.0	0.000	0.333333	0.030726	0.0	0.0	0.0	0.0	1.0
630	0.5	0.363636	0.5	0.250	0.000000	0.143462	0.0	0.5	0.5	0.0	0.0
50	1.0	0.386364	0.5	0.000	0.000000	0.014932	1.0	0.5	0.5	1.0	0.0

In [474]:

```
X_test.head()
```

Out[474]:

	pclass	sex	age	sibsp	parch	fare	embarked	who	adult_male	deck	alive
378	1.0	0.420455	0.5	0.125	0.000000	0.030937	0.0	0.5	0.5	0.0	0.0
101	1.0	0.000000	0.5	0.000	0.000000	0.015127	1.0	0.5	0.5	1.0	0.0
631	1.0	0.000000	0.5	0.000	0.000000	0.015412	1.0	0.5	0.5	1.0	0.0
175	1.0	0.420455	0.5	0.000	0.000000	0.018543	1.0	0.5	0.5	1.0	0.0
77	0.0	0.318182	0.5	0.000	0.166667	0.150855	0.0	0.5	0.5	0.0	0.0

In [475]:

```
y_train
```

Out[475]:

```
736    34.3750
181    15.0500
381    15.7417
655    73.5000
75      7.6500
        ...
860    14.1083
217    27.0000
654     6.7500
584     8.7125
709    15.2458
Name: fare, Length: 692, dtype: float64
```

In [476]:

```
y_test
```

Out[476]:

```
403    15.8500
126     7.7500
656     7.8958
200     9.5000
102    77.2875
        ...
427    26.0000
64     27.7208
286     9.5000
280     7.7500
43     41.5792
Name: fare, Length: 174, dtype: float64
```

In [477]:

```
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
```

In [478]:

```
lr.fit(X_train,y_train)
```

Out[478]:

```
LinearRegression()
```

In [479]:

```
#Test the model
y_pred = lr.predict(X_test)
y_pred
```

Out[479]:

```
array([ 1.58500000e+01,  7.75000000e+00,  7.89580000e+00,  9.50000000e+00,
        7.72875000e+01,  1.30000000e+01,  1.30000000e+01,  7.62920000e+00,
        2.91250000e+01,  7.75000000e+00,  1.05000000e+01,  2.30000000e+01,
        1.26500000e+01,  5.50000000e+01,  5.14792000e+01,  7.62917000e+01,
        7.65000000e+00,  2.65500000e+01,  7.75000000e+00,  1.87875000e+01,
        3.00000000e+01,  2.60000000e+01,  6.93000000e+01,  1.58500000e+01,
        8.40420000e+00,  7.75000000e+00,  2.60000000e+01,  5.64958000e+01,
        7.05000000e+00,  7.22500000e+00,  2.60000000e+01,  7.52080000e+00,
        8.36250000e+00,  1.05000000e+01,  1.44542000e+01,  5.64958000e+01,
        8.05000000e+00,  7.92500000e+00,  1.34500000e+02,  7.92000000e+01,
        3.90000000e+01,  7.89580000e+00,  1.30000000e+01,  1.30000000e+01,
        8.66250000e+00,  9.00000000e+00,  9.22500000e+00,  7.75000000e+00,
        1.30000000e+01,  1.30000000e+01,  1.87500000e+01,  7.77500000e+00,
        7.05420000e+00,  1.95000000e+01,  7.67292000e+01,  7.22500000e+00,
        3.00708000e+01,  1.45000000e+01,  7.75000000e+00,  7.74170000e+00,
        1.45000000e+01,  1.12417000e+01,  3.35000000e+01,  1.05000000e+01,
        2.85000000e+01,  7.85420000e+00,  2.79000000e+01,  7.05000000e+00,
        8.05000000e+00,  1.30000000e+01, -1.12889967e-13,  3.12750000e+01,
        7.89580000e+00,  1.01708000e+01, -1.12889967e-13,  9.50000000e+00,
        8.13750000e+00,  7.82920000e+00,  3.30000000e+01,  1.64866700e+02,
        7.73750000e+00,  9.21670000e+00,  2.97000000e+01,  2.62875000e+01,
        2.54667000e+01,  2.60000000e+01,  8.43330000e+00,  2.23583000e+01,
        1.15000000e+01,  1.30000000e+01,  1.22750000e+01,  3.96875000e+01,
        7.75000000e+00,  1.30000000e+01,  8.51670000e+00,  7.49580000e+00,
        3.55000000e+01,  8.05000000e+00,  7.92000000e+01,  7.77500000e+00,
        5.18625000e+01,  7.85420000e+00,  7.85420000e+00,  2.77208000e+01,
        2.65500000e+01,  7.72920000e+00,  1.52458000e+01,  7.25000000e+00,
        3.00000000e+01,  7.05000000e+00,  7.75000000e+00,  2.63000000e+02,
        1.30000000e+01,  7.22920000e+00,  8.21708000e+01,  2.05250000e+01,
        2.77500000e+01,  2.10750000e+01,  8.91042000e+01,  1.35000000e+01,
        2.60000000e+01,  2.60000000e+01,  8.05000000e+00,  7.92500000e+00,
        5.25542000e+01,  3.90000000e+01,  7.89580000e+00,  3.96000000e+01,
        1.28750000e+01,  1.24750000e+01,  8.05000000e+00,  7.25000000e+00,
        2.40000000e+01,  5.20000000e+01,  2.60000000e+01,  2.25250000e+01,
        5.20000000e+01,  7.25000000e+00,  7.89580000e+00,  7.85420000e+00,
        2.79000000e+01,  1.51550000e+02,  5.64958000e+01,  6.66000000e+01,
        7.12833000e+01,  2.10000000e+01,  1.23500000e+01,  1.05000000e+01,
        1.58500000e+01,  1.30000000e+01,  6.49580000e+00,  2.05250000e+01,
        1.92583000e+01,  5.50000000e+01,  2.11500000e+02,  7.82920000e+00,
        2.41500000e+01,  9.35000000e+00, -1.12889967e-13,  8.05000000e+00,
        1.55000000e+01,  8.05000000e+00,  9.58750000e+00,  2.02125000e+01,
        2.97000000e+01,  1.05000000e+01,  1.52458000e+01,  1.80000000e+01,
        7.22920000e+00,  2.60000000e+01,  2.77208000e+01,  9.50000000e+00,
        7.75000000e+00,  4.15792000e+01])
```

In [480]:

```
y_pred
```

Out[480]:

```
array([ 1.58500000e+01,  7.75000000e+00,  7.89580000e+00,  9.50000000e+00,
        7.72875000e+01,  1.30000000e+01,  1.30000000e+01,  7.62920000e+00,
        2.91250000e+01,  7.75000000e+00,  1.05000000e+01,  2.30000000e+01,
        1.26500000e+01,  5.50000000e+01,  5.14792000e+01,  7.62917000e+01,
        7.65000000e+00,  2.65500000e+01,  7.75000000e+00,  1.87875000e+01,
        3.00000000e+01,  2.60000000e+01,  6.93000000e+01,  1.58500000e+01,
        8.40420000e+00,  7.75000000e+00,  2.60000000e+01,  5.64958000e+01,
        7.05000000e+00,  7.22500000e+00,  2.60000000e+01,  7.52080000e+00,
        8.36250000e+00,  1.05000000e+01,  1.44542000e+01,  5.64958000e+01,
        8.05000000e+00,  7.92500000e+00,  1.34500000e+02,  7.92000000e+01,
        3.90000000e+01,  7.89580000e+00,  1.30000000e+01,  1.30000000e+01,
        8.66250000e+00,  9.00000000e+00,  9.22500000e+00,  7.75000000e+00,
        1.30000000e+01,  1.30000000e+01,  1.87500000e+01,  7.77500000e+00,
        7.05420000e+00,  1.95000000e+01,  7.67292000e+01,  7.22500000e+00,
        3.00708000e+01,  1.45000000e+01,  7.75000000e+00,  7.74170000e+00,
        1.45000000e+01,  1.12417000e+01,  3.35000000e+01,  1.05000000e+01,
        2.85000000e+01,  7.85420000e+00,  2.79000000e+01,  7.05000000e+00,
        8.05000000e+00,  1.30000000e+01, -1.12889967e-13,  3.12750000e+01,
        7.89580000e+00,  1.01708000e+01, -1.12889967e-13,  9.50000000e+00,
        8.13750000e+00,  7.82920000e+00,  3.30000000e+01,  1.64866700e+02,
        7.73750000e+00,  9.21670000e+00,  2.97000000e+01,  2.62875000e+01,
        2.54667000e+01,  2.60000000e+01,  8.43330000e+00,  2.23583000e+01,
        1.15000000e+01,  1.30000000e+01,  1.22750000e+01,  3.96875000e+01,
        7.75000000e+00,  1.30000000e+01,  8.51670000e+00,  7.49580000e+00,
        3.55000000e+01,  8.05000000e+00,  7.92000000e+01,  7.77500000e+00,
        5.18625000e+01,  7.85420000e+00,  7.85420000e+00,  2.77208000e+01,
        2.65500000e+01,  7.72920000e+00,  1.52458000e+01,  7.25000000e+00,
        3.00000000e+01,  7.05000000e+00,  7.75000000e+00,  2.63000000e+02,
        1.30000000e+01,  7.22920000e+00,  8.21708000e+01,  2.05250000e+01,
        2.77500000e+01,  2.10750000e+01,  8.91042000e+01,  1.35000000e+01,
        2.60000000e+01,  2.60000000e+01,  8.05000000e+00,  7.92500000e+00,
        5.25542000e+01,  3.90000000e+01,  7.89580000e+00,  3.96000000e+01,
        1.28750000e+01,  1.24750000e+01,  8.05000000e+00,  7.25000000e+00,
        2.40000000e+01,  5.20000000e+01,  2.60000000e+01,  2.25250000e+01,
        5.20000000e+01,  7.25000000e+00,  7.89580000e+00,  7.85420000e+00,
        2.79000000e+01,  1.51550000e+02,  5.64958000e+01,  6.66000000e+01,
        7.12833000e+01,  2.10000000e+01,  1.23500000e+01,  1.05000000e+01,
        1.58500000e+01,  1.30000000e+01,  6.49580000e+00,  2.05250000e+01,
        1.92583000e+01,  5.50000000e+01,  2.11500000e+02,  7.82920000e+00,
        2.41500000e+01,  9.35000000e+00, -1.12889967e-13,  8.05000000e+00,
        1.55000000e+01,  8.05000000e+00,  9.58750000e+00,  2.02125000e+01,
        2.97000000e+01,  1.05000000e+01,  1.52458000e+01,  1.80000000e+01,
        7.22920000e+00,  2.60000000e+01,  2.77208000e+01,  9.50000000e+00,
        7.75000000e+00,  4.15792000e+01])
```

In [487]:

```
from sklearn.metrics import r2_score
acc = r2_score(y_pred,y_test)
acc
```

Out[487]:

```
1.0
```

In [482]:

```
y_test
```

Out[482]:

```
403    15.8500
126     7.7500
656     7.8958
200     9.5000
102    77.2875
        ...
427    26.0000
64     27.7208
286     9.5000
280     7.7500
43     41.5792
Name: fare, Length: 174, dtype: float64
```

In [486]:

```
E = y_pred -  y_test
E
```

Out[486]:

```
403   -2.309264e-14
126    8.881784e-14
656   -1.429967e-13
200   -6.572520e-14
102   -1.421085e-13
           ...
427   -7.105427e-15
64     4.263256e-14
286   -3.907985e-14
280    2.611245e-13
43    -5.684342e-14
Name: fare, Length: 174, dtype: float64
```

In [ ]:

```

```

In [488]:

```
from sklearn.metrics import r2_score
r2_score(pred,y_test)*100
```

Out[488]:

```
29.59803876819417
```

In [ ]:

```

```

In [ ]:

```
```